```
Set     Items   Description
S1     247258   (MEMOR? OR BUFFER?)(3N)STACK? OR LIFO OR LAST()IN()FIRST()-
             OUT OR FIRST()IN(2N)LAST()OUT OR STACK?
S2       5764   S1(5N)(SECURE OR SECURES OR SECURING OR SECURED OR SECURITY
              OR PROTECT? OR GUARD? OR SHIELD? OR FORTIF? OR PREVENT?(2N)(-
             CORRUPT? OR OVERFLOW? OR UNDERFLOW? OR CRASH? OR SMASH?))
S3    1568621   (PROVID? OR SUPPLY? OR SUPPLIE? ? OR FURNISH? OR ADD OR AD-
             DS OR ADDING OR INPUT? OR INSERT? OR INCLU? OR ADDITION?)(5N)-
             (INFORMATION? OR BIT OR BITS OR BINARY()DIGIT? OR DATA OR VAL-
             UE? ? OR ENTRY? OR ENTRIES? OR NUMBER? OR GUARD(2N)VARIABL?)
S4       6125   S1(5N)(BEFORE? OR PRIOR OR AHEAD OR IN()ADVANCE OR PREVIOU-
             S? OR SUBSEQUEN? OR EARLIER? OR ALREADY? OR PRECED?)
S5      16418   S1(7N)(NEXT OR LATER? OR FOLLOW? OR ENSU??? OR AFTER OR CO-
            MING OR SUBSEQUEN? OR CONSEQUENT? OR FORTHCOMING)
S6     248873   (PERFORM? OR EXECUT? OR IMPLEMENT? OR OPERATE? ? OR OPERAT-
             ING OR ENACT? OR HANDL? OR (CARRY? OR CARRIE? ?)()OUT OR COMP-
             LET? OR ENABL? OR ALLOW?)(5N)(FUNCTION? OR COMMMAND? OR INSTR-
             UCT? OR ALLOCAT? OR RE()ALLOCAT? OR DE()ALLOCAT?)
S7         238   S1:S2 AND S3 AND S4 AND S5
S8          49   S6 AND S7
S9          13   S7 AND S2 AND S3 AND S4 AND S5
S10         13   S2 AND S3 AND S4 AND S5
S11        238   S3 AND S4 AND S5
S12        476   S11 AND (MEMOR? OR BUFFER?)(3N)STACK? OR LIFO OR LAST()IN(-
             )FIRST()OUT OR FIRST()IN(2N)LAST()OUT
S13         79   S12 AND S6
S14         76   S13 NOT S10
S15          2   S14 AND (SECURE OR SECURES OR SECURING OR SECURED OR SECUR-
             ITY OR PROTECT? OR GUARD? OR SHIELD? OR FORTIF? OR PREVENT?(2-
             N)(CORRUPT? OR OVERFLOW? OR UNDERFLOW? OR CRASH? OR SMASH?))
S16        469   S12 NOT (S10 OR S15)
S17         22   S16 AND (SECURE OR SECURES OR SECURING OR SECURED OR SECUR-
             ITY OR PROTECT? OR GUARD? OR SHIELD? OR FORTIF? OR PREVENT?(2-
             N)(CORRUPT? OR OVERFLOW? OR UNDERFLOW? OR CRASH? OR SMASH?))
File 350:Derwent WPIX 1963-2007/UD=200745
         (c) 2007 The Thomson Corporation
File 347:JAPIO Dec 1976-2007/Dec(Updated 070702)
         (c) 2007 JPO & JAPIO
```

* Date may not be good

9/69,K/7 **(Item 7 from file: 350)**
DIALOG(R)File 350:Derwent WPIX


0009606854 - Drawing available
WPI ACC NO: 1999-556443/199947
XRPX Acc No: N1999-412316
**Microprocessor - has execution circuit which resembles parallel branch to subroutine and** security **stack , when first instruction is read-out from instruction memory**
Patent Assignee: MATSUSHITA DENKI SANGYO KK (MATU); MATSUSHITA ELECTRIC IND CO LTD (MATU)
Inventor: TAKAYAMA S; TANAKA T
**Patent Family** (3 patents, 2 countries)

| Patent Number | Kind | Date | Application Number | Kind | Date | Update | |
|---|---|---|---|---|---|---|---|
| JP 11242595 | A | 19990907 | JP 1998348012 | A | 19981208 | 199947 | B |
| JP 3079090 | B2 | 20000821 | JP 1998348012 | A | 19981208 | 200043 | E |
| US 6212630 | B1 | 20010403 | US 1998208200 | A | 19981209 | 200120 | E |

Priority Applications (no., kind, date): JP 1997339669 A 19971210

**Patent Details**

| Number | Kind | Lan | Pg | Dwg | Filing Notes |
|---|---|---|---|---|---|
| JP 11242595 | A | JA | 25 | 40 | |
| JP 3079090 | B2 | JA | 24 | | Previously issued patent JP 11242595 |

**Alerting Abstract** JP A
NOVELTY - A subroutine call circuit arranges the order of a subroutine branch execution. A **stack securing** circuit provides a new **stack** area into a data memory (38) for the next operation. An execution circuit reassemble the parallel branch to subroutine and **stack security** , when a first instruction is read-out from an instruction memory (39). DETAILED DESCRIPTION - An INDEPENDENT CLAIM is included for the programming language compiler.
USE - None given.
ADVANTAGE - Shortens execution time by performing sequential reading of subroutine branch, **stack security** and reset address, eliminating the independent **stack security** for evacuation of reset address and performing parallel operation. Prevents damage of reset address by subroutine since decompression of original reset address is performed.
DESCRIPTION OF DRAWING(S) - The figure shows the explanatory block diagram of decipherment execution of absolute instruction of the processor. (38) Data memory; (39) Instruction memory.

**Title Terms**/Index Terms/Additional Words: MICROPROCESSOR; EXECUTE; CIRCUIT; RESEMBLE; PARALLEL; BRANCH; SUBROUTINES; SECURE; **STACK** ; FIRST; INSTRUCTION; READ-OUT; MEMORY

**Class Codes**
International Classification (Main): G06F-009/40, G06F-009/42
(Additional/Secondary): G06F-009/45
US Classification, Issued: 712242000, 717005000

File Segment: EPI;
DWPI Class: T01
Manual Codes (EPI/S-X): T01-F04; T01-F05A

**...has execution circuit which resembles parallel branch to subroutine and**

security stack , **when first instruction is read-out from instruction memory**

**Original Titles:**
...Microprocessor for overlapping **stack** frame allocation with saving of subroutine data into **stack** area.

**Alerting Abstract** ...NOVELTY - A subroutine call circuit arranges the order of a subroutine branch execution. A **stack securing** circuit provides a new **stack** area into a data memory (38) for the next operation. An execution circuit reassemble the parallel branch to subroutine and **stack security** , when a first instruction is read-out from an instruction memory (39). DETAILED DESCRIPTION - An...

...ADVANTAGE - Shortens execution time by performing sequential reading of subroutine branch, **stack security** and reset address, eliminating the independent **stack security** for evacuation of reset address and performing parallel operation. Prevents damage of reset address by...

**Title Terms**.../Index Terms/Additional Words: **STACK** ;

**Original Publication Data by Authority**

**Original Abstracts:**
...in the LR 13 as a return address; (2) a branch operation for storing the **entry** address of the subroutine **included** in the subroutine call instruction in **the** PC 15; and (3) **a stack** reserve operation for preparing for the **following** use of a **stack** area by **adding** a **value** "-4" to the **value stored** in the SP **12** using the **adder** 22and **for** storing the **addition result** in the SP 12. 
**Claims:**
A microprocessor which is connected to a **data memory providing a stack** area **and** to an instruction **memory** prestoring instructions, comprising: a subroutine call means for causing a branch to a subroutine in an execution sequence; a **stack** reserve means for reserving a **new stack** area in the data **memory by** updating a **stack** pointer **using** a fixed value; **an** instruction fetch means for fetching an instruction from the instruction memory; a first instruction execution...
...fetched instruction is a first instruction, the subroutine call means cause the branch and the **stack** reserve means reserve the new **stack** area in parallel; **and** a post-branch data save **means** for having, **after** the branch by the subroutine call means, the **stack** reserve means **reserve** another new **stack** area in parallel with a **saving** of data into the **previously reserved** new **stack** area.

9/9/13 (Item 1 from file: 347)
DIALOG(R)File 347:JAPIO


04785492 **Image available**
**STACK** SWITCHING CONTROL UNIT

PUB. NO.: 07-078092 [JP 7078092 A]
PUBLISHED: March 20, 1995 (19950320)
INVENTOR(s): OISHI SHOHEI
APPLICANT(s): MATSUSHITA ELECTRIC IND CO LTD [000582] (A Japanese Company or Corporation), JP (Japan)
APPL. NO.: 05-223249 [JP 93223249]
FILED: September 08, 1993 (19930908)
INTL CLASS: [6] G06F-009/46; G06F-009/46
JAPIO CLASS: 45.1 (INFORMATION PROCESSING -- Arithmetic Sequence Units)

ABSTRACT

PURPOSE: To enable respective tasks to share one **stack** area and to reduce capacity that should be **secured** as the **stack** area by **providing** a **stack data** movement control part which can move **stack** data of respective tasks.

CONSTITUTION: A task 1 is being actuated **before** task switching, so **stack** data 11 corresponding to this task 1 are adjacent to a common free area 5a. A **stack** pointer 4 indicates the address 6 of the border between the common free area 5a and data 11 and data are put in and out of there. Namely, when data are put in, the data are taken in the area 5a and regarded as part of data 11. When the data are taken out, the data are taken out of the data 11. Consequently, the area 5a becomes large. Further, a task 3 becomes a start task **after** the switching, so **stack** data 13 corresponding to the task 3 adjoin to the area 5a and data can be taken in and out of the data 13.

**15/69,K/2 (Item 2 from file: 350)**
DIALOG(R)File 350:Derwent WPIX


0004372776 - Drawing available
WPI ACC NO: 1988-106857/198816
**Buffer address register for computer system - has several address input ports and overlapping loading and reading operations to speed up address retrieval and storage**
Patent Assignee: BULL HN INFORMATION SYSTEMS INC (HONE); HONEYWELL BULL INC (HONE)
Inventor: IZBICKI K J; LEMAY R A; TAGUE S A; WOODS W E
**Patent Family** (4 patents, 7 countries)

| Patent Number | Kind | Date | Application Number | Kind | Date | Update |
|---|---|---|---|---|---|---|
| EP 264077 | A | 19880420 | EP 1987114769 | A | 19871009 | 198816 B |
| AU 198779734 | A | 19880421 | | | | 198824 E |
| CA 1293333 | C | 19911217 | | | | 199206 E |
| US 5161217 | A | 19921103 | US 1986918227 | A | 19861014 | 199247 E |
| | | | US 1989300732 | A | 19890123 | |
| | | | US 1989418084 | A | 19891006 | |

Priority Applications (no., kind, date): US 1989418084 A 19891006; US 1989300732 A 19890123; US 1986918227 A 19861014
**Patent Details**

| Number | Kind | Lan | Pg | Dwg | Filing Notes |
|---|---|---|---|---|---|
| EP 264077 | A | EN | 13 | | |
| Regional Designated States,Original: DE FR GB IT | | | | | |
| CA 1293333 | C | EN | | | |
| US 5161217 | A | EN | 11 | 4 | Continuation of application US 1986918227 |
| | | | | | Continuation of application US 1989300732 |

**Alerting Abstract** EP A
The address register has several registers for temporarily storing addresses received from several address sources. A pop-stack register stores several addresses. Addresses stored in the registers are transferred to and stored in the pop-stack register if the addresses are not read out and used for reading instructions from the memory before another address is stored in one of the registers. A local register to which addresses stored in the pop-stack register are individually transferred and stored for subsequent read out, to address the memory for instructions. One of the registers has an interrupt register which receives and stores an interrupt address from the computer system upon an error being detected whereupon the processing is to be interrupted. The interrupt address indicates the memory address of the next program **instruction** to be **executed** following correction of the error that caused the interrupt.

**Equivalent Alerting Abstract** US A
The **last - in first - out** register has multiple address input ports and stores a number of addresses. Addresses loading operations are over-lapped with address reading operations to speed up the rate at which address may be stored in and retrieved from the register.
When the register is full of address it provides an indication which permits; the addresses already stored in the register to be read out and stored in an external memory, then additional addresses to be stored in the register, and subsequently the addresses transferred to the memory for storage to be transferred to the buffer address register for read out.
USE - Buffer address register in computer system.

**Title Terms**/Index Terms/Additional Words: BUFFER; ADDRESS; REGISTER; COMPUTER; SYSTEM; INPUT; PORT; OVERLAP; LOAD; READ; OPERATE; SPEED; UP; RETRIEVAL; STORAGE

**Class Codes**
International Classification (Main): G06F-009/22
(Additional/Secondary): G06F-012/00, G06F-005/06, G06F-007/00, G06F-009/32
US Classification, Issued: 395375000, 364DIG001, 364238600, 364247000, 395400000, 395425000

File Segment: EPI;
DWPI Class: T01
Manual Codes (EPI/S-X): T01-F; T01-H01D

**Original Titles:**
...Buffered address stack register with parallel input registers and overflow **protection**

**Alerting Abstract** ...is to be interrupted. The interrupt address indicates the memory address of the next program **instruction** to be **executed** following correction of the error that caused the interrupt.

**Equivalent Alerting Abstract** ...The **last - in first - out** register has multiple address input ports and stores a number of addresses. Addresses loading operations...

**Original Publication Data by Authority**

**Original Abstracts:**
...A **last - in , first - out register having multiple** address input ports and capable of storing a plurality of addresses. Address loading operations are...
**Claims:**
...is to be interrupted. The interrupt address indicates the memory address of the next program **instruction** to be **executed** following correction of the error that caused the interrupt.
?

**17/69,K/7 (Item 7 from file: 350)**
DIALOG(R)File 350:Derwent WPIX


0009325034 - Drawing available
WPI ACC NO: 1999-256802/199922
XRPX Acc No: N1999-191364
**Generating internal crypto-keys**
Patent Assignee: NEC CORP (NIDE)
Inventor: SHIMADA M
**Patent Family** (4 patents, 27 countries)

| Patent Number | Kind | Date | Application Number | Kind | Date | Update | |
|---|---|---|---|---|---|---|---|
| EP 913964 | A2 | 19990506 | EP 1998120404 | A | 19981028 | 199922 | B |
| JP 11136229 | A | 19990521 | JP 1997314567 | A | 19971031 | 199931 | E |
| JP 3092567 | B2 | 20000925 | JP 1997314567 | A | 19971031 | 200051 | E |
| US 6278780 | B1 | 20010821 | US 1998182014 | A | 19981029 | 200150 | E |

Priority Applications (no., kind, date): JP 1997314567 A 19971031

**Patent Details**

| Number | Kind | Lan | Pg | Dwg | Filing Notes |
|---|---|---|---|---|---|
| EP 913964 | A2 | EN | 13 | 6 | |
| Regional Designated States,Original: AL AT BE CH CY DE DK ES FI FR GB GR IE IT LI LT LU LV MC MK NL PT RO SE SI | | | | | |
| JP 11136229 | A | JA | 8 | | |
| JP 3092567 | B2 | JA | 9 | | Previously issued patent JP 11136229 |

**Alerting Abstract** EP A2
NOVELTY - Method involves outputting m sets of the first conversion results, each i-th of the m sets being obtained by processing an (i-1)th of the m sets with a first nonlinear function; where m is a positive integer > 1, m > i > 1. m sets of second conversion results are output by processing the second part of the external key with a second nonlinear function. Each j-th of the m internal crypto-keys is output by combining the j-th of the m sets of the first conversion results and the (m-j+1)th of the m sets of the second according to a combining function, j < m.
USE - For generating internal crypto-keys used as initial values to be set in feedback registers of pseudo-random sequence generator. These are used for eXclusive OR logic added to a data sequence recorded in recording medium or transmitted in communication system to prevent third party tapping data sequence without permission.
ADVANTAGE - Allows rapid and **secure** setting of internal crypto-keys in feedback shift registers.
DESCRIPTION OF DRAWINGS - The drawing shows a functional block diagram illustrating an apparatus for generating the internal crypto-keys.

**Title Terms**/Index Terms/Additional Words: GENERATE; INTERNAL; KEY

**Class Codes**
International Classification (Main): H04L-009/00, H04L-009/08, H04L-009/26
(Additional/Secondary): H04L-009/20
US Classification, Issued: 380047000, 380044000, 380278000, 380264000, 380223000

File Segment: EPI;
DWPI Class: W01
Manual Codes (EPI/S-X): W01-A05A

**Alerting Abstract** ...ADVANTAGE - Allows rapid and **secure** setting of internal crypto-keys in feedback shift registers...

**Original Publication Data by Authority**

**Original Abstracts:**
...shift-registers of a pseudo-random-sequence generator of a stream cipher system with sufficient **security** and sufficiently high **speed** as well, the method comprises: a step of outputting m sets of first conversion results...

...shift-registers of a pseudo-random-sequence generator of a stream cipher system with sufficient **security** and sufficiently high speed **as** well, the method comprises: a step of outputting m sets of first conversion results, obtaining...

**Claims:**
...the input bit sequence to be processed by the second one-way-function circuit;a **LIFO** ( **Last - In - First - Out** ) buffer wherein conversion results outputted from the second one-way-function circuit are stacked in synchronization with the clock signal when **the LIFO buffer is controlled** in a writing mode, and the conversion results stacked in the **LIFO** buffer are popped up in synchronization with the clock signal **when** the **LIFO** buffer is controlled in a reading mode; anda combining circuit for **outputting** internal crypto-keys in synchronization with the clock signal by combining outputs **of** the **LIFO** buffer and the first one-way-function circuit.

**17/69,K/10 (Item 10 from file: 350)**
DIALOG(R)File 350:Derwent WPIX


0002429560
WPI ACC NO: 1982-A0752J/198247
**Error recovery system for programmable data processor - uses cast in-first out stack following execution and periodic error tests to use stack for error recovery if needed**
Patent Assignee: GRANDJEAN B (GRAN-I)
Inventor: GRANDJEAN B; KUBIAK C
**Patent Family** (1 patents, 1 countries)

| Patent Number | Kind | Date | Application Number | Kind | Date | Update |
|---|---|---|---|---|---|---|
| FR 2503900 | A | 19821015 | FR 19817410 | A | 19810413 | 198247 B |

**Patent Details**

| Number | Kind | Lan | Pg | Dwg | Filing Notes |
|---|---|---|---|---|---|
| FR 2503900 | A | FR | 25 | 4 | |

**Alerting Abstract** FR A
The arrangement uses a memory to hold the current state of the data during program execution. A **security** memory (16), organised in a **last in first out** stack configuration is provided together with a circuit (17) to copy into the **security** memory the address and previous state of each data element in a memory (14) before changing the memory state by writing to it. A system is provided for the creation of recovery points and a fixed target location for the copied program.
An error flag is periodically tested and initiates return to a recovery point, setting the stack (16) pointer to the last **secure** entry, while a clear flag empties the stack for refilling.

**Title Terms**/Index Terms/Additional Words: ERROR; RECOVER; SYSTEM; PROGRAM; DATA; PROCESSOR; CAST; FIRST; STACK; FOLLOW; EXECUTE; PERIODIC; TEST; NEED

**Class Codes**
International Classification (+ Attributes)

| IPC + Level Value | Position | Status | Version |
|---|---|---|---|
| G06F-0011/14 A | I | R | 20060101 |
| G06F-0011/14 C | I | R | 20060101 |

File Segment: EPI;
DWPI Class: T01
Manual Codes (EPI/S-X): T01-G02; T01-G09
**Alerting Abstract** ...uses a memory to hold the current state of the data during program execution. A **security** memory (16), organised in a **last in first out** stack configuration is provided together with a circuit (17) to copy into the **security** memory the address and previous state of each data element in a memory (14) before...

...and initiates return to a recovery point, setting the stack (16) pointer to the last **secure** entry, while a clear flag empties the stack for refilling.

**17/9/13 (Item 3 from file: 347)**
DIALOG(R)File 347:JAPIO


03819449 **Image available**
INFORMATION PROCESSOR

PUB. NO.: 04-184549 [JP 4184549 A]
PUBLISHED: July 01, 1992 (19920701)
INVENTOR(s): HIRONAKA MASA
APPLICANT(s): NEC CORP [000423] (A Japanese Company or Corporation), JP (Japan)
APPL. NO.: 02-312885 [JP 90312885]
FILED: November 20, 1990 (19901120)
INTL CLASS: [5] G06F-012/08; G06F-012/12
JAPIO CLASS: 45.2 (INFORMATION PROCESSING -- Memory Units)
JAPIO KEYWORD: R131 (INFORMATION PROCESSING -- Microcomputers & Microprocessers)
JOURNAL: Section: P, Section No. 1440, Vol. 16, No. 506, Pg. 55, October 20, 1992 (19921020)

ABSTRACT

PURPOSE: To prevent a decline in the hitting rate of a cache memory due to numerous issuance of write access to a write **protect** area by temporarily holding hit data in the cache **memory** in a **stack** .

CONSTITUTION: When the data section 13 of a cache memory recognizes that the cache memory is hit, the section 13 outputs data A to a data bus 25 and stores the data B on another data bus 23 in the area from which the data A are outputted **after** fetching the data B. A **stack** 14 fetches and holds the **data** A. When the address **inputted** to a memory controller 3 is that of the write **protect** area of a main memory 2, a comparator 15 makes the **stack** 14 to drive the **previously** holding data A to the data bus 25 and the data section 13 to store the data A in the area where the data A are previously replaced with the data B. Therefore, a decline in the hitting rate of the cache memory due to numerous issuance of write access to the write **protect** area can be prevented, since invalidation of hit data is not required.

**17/9/20 (Item 10 from file: 347)**
DIALOG(R)File 347:JAPIO


00490947
**LAST - IN FIRST - OUT** MEMORY

PUB. NO.: 54-142947 [JP 54142947 A]
PUBLISHED: November 07, 1979 (19791107)
INVENTOR(s): YANO MASAAKI
OUCHI YASUNORI
MATSUHIRO KAZUYOSHI
APPLICANT(s): NEC CORP [000423] (A Japanese Company or Corporation), JP (Japan)
NIPPON TELEGR & TELEPH CORP <NTT> [000422] (A Japanese Company or Corporation), JP (Japan)
APPL. NO.: 53-052131 [JP 7852131]
FILED: April 27, 1978 (19780427)
INTL CLASS: [2] G11C-007/00; G06F-009/16
JAPIO CLASS: 45.2 (INFORMATION PROCESSING -- Memory Units); 45.1 (INFORMATION PROCESSING -- Arithmetic Sequence Units)
JOURNAL: Section: E, Section No. 163, Vol. 04, No. 1, Pg. 153, January 08, 1980 (19800108)

ABSTRACT

PURPOSE: To **secure** an immediate return to the preceding state from the interruption process routine by setting up the independent address and the address grown at the address growing circuit based on the address setting signals given from outside.

CONSTITUTION: Memory circuit 20 gives the first reading to the data supplied last, and address growing circuit 26 renews the addresses in sequence in order to grow address 23 showing the storage position of circuit 20. Renewed address storing circuit 27 stores address 29 given from circuit 26 and then applies it to circuit 26 again. When set operation designation signal 24-3 freatures ''1'', the contents supplied from outside via input port 28 is written into circuit 27. At the same time, the contents of circuit 20 of the address designated by the contents applied to circuit 27 is read out in the form of output data 25. Thus, an immediate return is possible.

```
Set     Items   Description
S1     167077   (MEMOR? OR BUFFER?)(3N)STACK? OR LIFO OR LAST()IN()FIRST()-
              OUT OR FIRST()IN(2N)LAST()OUT OR STACK?
S2       5163   S1(5N)(SECURE OR SECURES OR SECURING OR SECURED OR SECURITY
               OR PROTECT? OR GUARD? OR SHIELD? OR FORTIF? OR PREVENT?(2N)(-
              CORRUPT? OR OVERFLOW? OR UNDERFLOW? OR CRASH? OR SMASH?))
S3     749685   (PROVID? OR SUPPLY? OR SUPPLIE? ? OR FURNISH? OR ADD OR AD-
              DS OR ADDING OR INPUT? OR INSERT? OR INCLU? OR ADDITION?)(5N)-
              (INFORMATION? OR BIT OR BITS OR BINARY()DIGIT? OR DATA OR VAL-
              UE? ? OR ENTRY? OR ENTRIES? OR NUMBER? OR GUARD(2N)VARIABL?)
S4      13450   S1(5N)(BEFORE? OR PRIOR OR AHEAD OR IN()ADVANCE OR PREVIOU-
              S? OR SUBSEQUEN? OR EARLIER? OR ALREADY? OR PRECED?)
S5      18928   S1(5N)(NEXT OR LATER? OR FOLLOW? OR ENSU??? OR AFTER OR CO-
              MING OR SUBSEQUEN? OR CONSEQUENT? OR FORTHCOMING)
S6     411045   (PERFORM? OR EXECUT? OR IMPLEMENT? OR OPERATE? ? OR OPERAT-
              ING OR ENACT? OR HANDL? OR (CARRY? OR CARRIE? ?)()OUT OR COMP-
              LET? OR ENABL? OR ALLOW?)(5N)(FUNCTION? OR COMMMAND? OR PROGR-
              AM? OR SUBROUTINE? OR INSTRUCT? OR ALLOCAT? OR RE()ALLOCAT? OR
               DE()ALLOCAT?
S7        1043   S1:S2(100N)S3(100N)S4(100N)S5
S8          69   S7(100N)S2(100N)S3(100N)S4(100N)S5
S9          36   S8 NOT (AD>1998 OR AD=1999:2007)
S10         13   S9(100N)((MEMOR? OR BUFFER?)(3N)STACK? OR LIFO OR LAST()IN-
              ()FIRST()OUT OR FIRST()IN(2N)LAST()OUT)
S11        108   S3(100N)S4(100N)S5(100N)((MEMOR? OR BUFFER?)(3N)STACK? OR -
              LIFO OR LAST()IN()FIRST()OUT OR FIRST()IN(2N)LAST()OUT)
S12         13   S11(100N)S2
S13          2   S12 NOT S10
File 348:EUROPEAN PATENTS 1978-2007/ 200729
         (c) 2007 European Patent Office
File 349:PCT FULLTEXT 1979-2007/UB=20070719UT=20070712
         (c) 2007 WIPO/Thomson
```

```
Set     Items   Description
S1     270900   (MEMOR? OR BUFFER?)(3N)STACK? OR LIFO OR LAST()IN()FIRST()-
             OUT OR FIRST()IN(2N)LAST()OUT OR STACK?
S2       1321   S1(7N)(SECURE OR SECURES OR SECURING OR SECURED OR SECURITY
              OR PROTECT? OR GUARD? OR SHIELD? OR FORTIF? OR PREVENT?(2N)(-
             CORRUPT? OR OVERFLOW? OR UNDERFLOW? OR CRASH? OR SMASH?))
S3    1689169   (PROVID? OR SUPPLY? OR SUPPLIE? ? OR FURNISH? OR ADD OR AD-
             DS OR ADDING OR INPUT? OR INSERT? OR INCLU? OR ADDITION?)(5N)-
             (INFORMATION? OR DATA OR VALUE? ? OR ENTRY? OR ENTRIES? OR NU-
             MBER? OR GUARD()VARIABL?)
S4       1825   S1(3N)(BEFORE? OR PRIOR OR AHEAD OR IN()ADVANCE OR PREVIOU-
             S? OR SUBSEQUEN? OR EARLIER? OR ALREADY? OR PRECED?)
S5       3976   S1(3N)(NEXT OR LATER? OR FOLLOW? OR ENSU??? OR AFTER OR CO-
             MING OR SUBSEQUEN? OR CONSEQUENT? OR FORTHCOMING)
S6     871105   (PERFORM? OR EXECUT? OR IMPLEMENT? OR OPERATE? ? OR OPERAT-
             ING OR ENACT? OR HANDL? OR (CARRY? OR CARRIE? ?)()OUT OR COMP-
             LET? OR ENABL? OR ALLOW? OR CALL???)(5N)(FUNCTION? OR COMMMAN-
             D? OR PROGRAM? OR SUBROUTINE? OR INSTRUCT? OR ALLOCAT? OR RE(-
             )ALLOCAT? OR D
S7         40   S1:S2 AND S3 AND S4 AND S5
S8          0   S7 AND S2 AND S3 AND S4 AND S5
S9         27   S7 NOT (PY>1998 OR PY=1999:2007)
S10        13   RD  (unique items)
File   2:INSPEC 1898-2007/Jul W2
         (c) 2007 Institution of Electrical Engineers
File   6:NTIS 1964-2007/Jul W4
         (c) 2007 NTIS, Intl Cpyrght All Rights Res
File   8:Ei Compendex(R) 1884-2007/Jul W3
         (c) 2007 Elsevier Eng.  Info. Inc.
File  34:SciSearch(R) Cited Ref Sci 1990-2007/Jul W4
         (c) 2007 The Thomson Corp
File  35:Dissertation Abs Online 1861-2007/Jul
         (c) 2007 ProQuest Info&Learning
File  56:Computer and Information Systems Abstracts 1966-2007/Jul
         (c) 2007 CSA.
File  60:ANTE: Abstracts in New Tech & Engineer 1966-2007/Jul
         (c) 2007 CSA.
File  62:SPIN(R) 1975-2007/Jul W2
         (c) 2007 American Institute of Physics
File  65:Inside Conferences 1993-2007/Jul 23
         (c) 2007 BLDSC all rts. reserv.
File  95:TEME-Technology & Management 1989-2007/Jul W3
         (c) 2007 FIZ TECHNIK
File  99:Wilson Appl. Sci & Tech Abs 1983-2007/Jun
         (c) 2007 The HW Wilson Co.
File 111:TGG Natl.Newspaper Index(SM) 1979-2007/Jul 16
         (c) 2007 The Gale Group
File 144:Pascal 1973-2007/Jul W3
         (c) 2007 INIST/CNRS
File 239:Mathsci 1940-2007/Sep
         (c) 2007 American Mathematical Society
File 256:TecInfoSource 82-2007/Aug
         (c) 2007 Info.Sources Inc
File 434:SciSearch(R) Cited Ref Sci 1974-1989/Dec
         (c) 2006 The Thomson Corp
File 583:Gale Group Globalbase(TM) 1986-2002/Dec 13
         (c) 2002 The Gale Group
```

**10/7/7 (Item 7 from file: 2)**
DIALOG(R)File 2:INSPEC


02417664 INSPEC Abstract Number: A79092612
**Title: Superstack-an iterative** stacking **algorithm**
Author(s): Naess, O.E.
Author Affiliation: Norwegian Petroleum Directorate, Stavanger, Norway
Journal: Geophysical Prospecting vol.27, no.1 p.16-28
Publication Date: March 1979 Country of Publication: Netherlands
CODEN: GPPRAR ISSN: 0016-8025
Language: English Document Type: Journal Paper (JP)
Treatment: Practical (P)
Abstract: An algorithm for separation of signals according to their coherency is proposed. The algorithm, called Superstack, is used on common depth point data after normal moveout corrections have been applied. The algorithm can be regarded as an iterative **stacking** procedure. **After** each **stack** , **input values** are changed depending on the consistency of the output of the **previous stack** . The Superstack algorithm is able to provide better separation of signals showing a different degree of horizontal consistency than the normal horizontal **stack** . (2 Refs)
Subfile: A